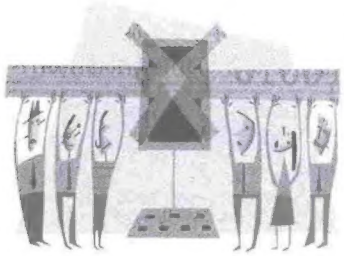
■ ภาษิต ภัทรมูล

SOFT MAIL



DelphiX กับ Delphi ตอนที่ 3

ยังคงอยู่ที่ การแสดงรูปภาพ แต่จะมีเนื้อหาเพิ่มเติม จากตอนที่แล้ว

บทความตอนนี้ยังคงอยู่ที่การแสดงรูปภาพ แต่จะมีเนื้อหาเพิ่มเติมจากตอนที่แล้ว โดยมีส่วนหลัก ๆ ดังนี้

1. การนำรูปภาพมาแสดงที่หน้าจอ
2. การใช้งาน Dximagelist กับแฟ้มรูปภาพสกุล .dxg ซึ่งสร้างจากโปรแกรม Imagelib
3. การโปรแกรม Animation 2 มิติโดยการสลับรูปภาพจากการสุ่ม
4. การเคลื่อนไหววัตถุ Sprite โดยการสุ่มตำแหน่ง
5. การใช้งาน Dxinput กับคีย์บอร์ดและเมาส์

ขณะนี้ DelphiX ได้ปรับเวอร์ชันใหม่แล้วครับ ปัญหาน้อยลง โดยเฉพาะ Help ที่ใช้กับ Windows Thai Edition ไม่ได้ก็สามารถเปิดอ่านได้แล้ว ใครสนใจก็ไปดาวน์โหลดที่ http://www.ingjapan.ne.jp/hori/ หรือใช้บริการของซอฟต์เมล์ซึ่งผู้เขียนแนบแฟ้มมาด้วยแล้ว (DelphiX991024.zip) โดยเฉพาะการใช้งานกับ Delphi 4.0 ก็ใช้ได้โดยติดตั้งธรรมดาไม่ต้องไปก๊อบนั่นก๊อบนี่เพิ่มเติมให้ปวดหัวอีกแล้ว

โปรเจ็กต์ตัวอย่าง

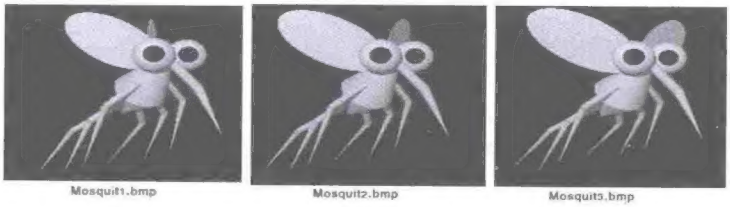
แสดงการจัดการกับรูปภาพ โดยมีความสามารถดังนี้

1. นำรูปภาพที่เตรียมไว้มาแสดงหน้าจอ โดยให้เปลี่ยนรูปภาพทุก ๆ ช่วงระยะเวลา เป็นลักษณะของการเคลื่อนไหวแบบง่าย
2. รูปภาพที่ปรากฏบนจอ สามารถเคลื่อนไหวได้ ตามการกดแป้นคีย์บอร์ดและการขยับเมาส์
3. แสดงรูปภาพที่ปรากฏบนจอโดยการสุ่มตำแหน่ง สามารถเคลื่อนไหวได้อย่างเป็นอิสระ

เตรียมการ

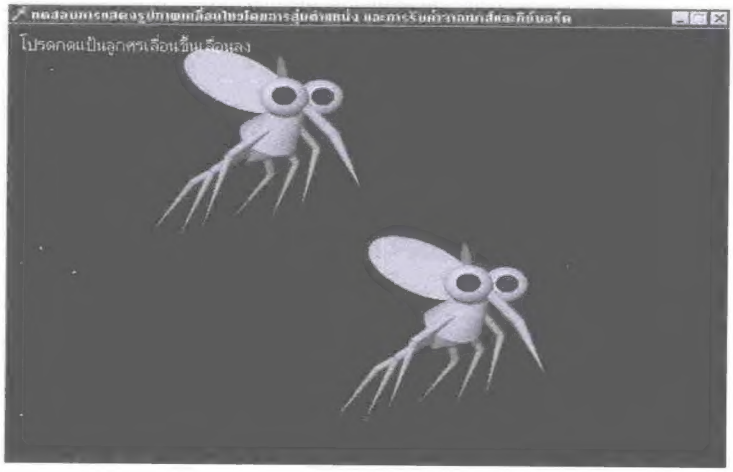
อันดับแรกเป็นการเตรียมรูปภาพเพื่อแสดงที่หน้าจอ รูปที่นำมาเป็นตัวอย่างนี้อยู่ใน Ms.DirectX SDK 6.0 จากตัวอย่างใน C:\mssdk\samples\Multimedia\DDraw\src\Mosquito โดยมีภาพ 3 ภาพเป็นภาพยุงซึ่งมีการเคลื่อนไหวในอริยาบทต่าง ๆ กัน 3 แบบ ดังรูปที่ 1 ภาพเหล่านี้เรานำมาใช้โดยการสร้างเป็นแฟ้ม .Dxg ขึ้น

หลักการของโปรเจ็กต์ตัวอย่าง ผู้เขียนยึดแนวคิดจากตัวอย่างใน Ms.DirectX SDK 6.0 ซึ่งเขียนโค้ดโดย Ms. Visual C++ เรานำมาแปลงเป็น Delphi โดยใช้ DelphiX ซึ่งขั้นตอนต่าง ๆ จะง่ายกว่ามากและดูมีโครงสร้างที่เป็นระเบียบเรียบร้อยกว่า



รูปที่ 1

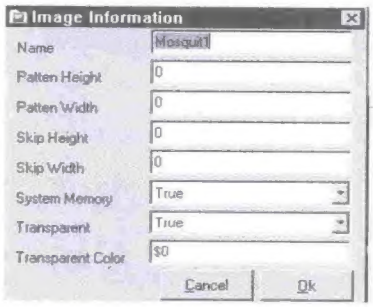
ในโปรเจ็กต์ตัวอย่างจะมียุงปรากฏขึ้นในหน้าจอ 2 ตัว ยุงตัวที่หนึ่งจะมีการเคลื่อนไหวโดยการเปลี่ยนอิริยาบทตามการสุ่มค่า เปลี่ยนตำแหน่งโดยรับค่าจากการกดคีย์บอร์ดและการขยับเมาส์ ส่วนยุงตัวที่สองจะมีการเคลื่อนไหวโดยการเปลี่ยนอิริยาบทตามการสุ่มค่า เปลี่ยนตำแหน่งจากการสุ่มค่า ให้เป็นภาพเคลื่อนไหวที่ดูต่อเนื่องเหมือนมีชีวิตจิตใจ ดังรูปที่ 2



รูปที่ 2

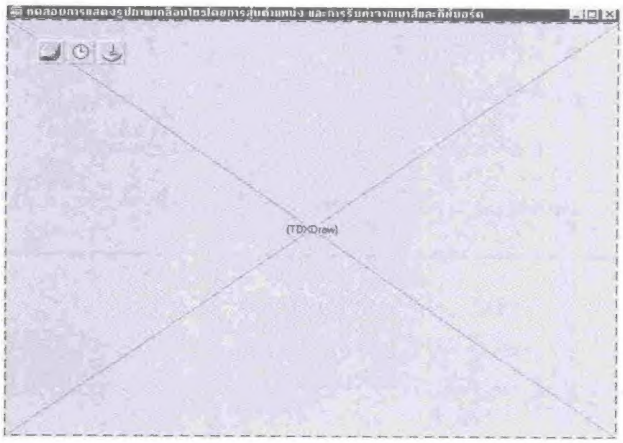
การสร้างแฟ้มสกุล .Dxg

เนื่องจากรูปภาพที่เรานำมาใช้นี้จะมีพื้นหลังเป็นสีดำ เราจึงต้องไปกำหนดในคุณสมบัติ Image Information หัวข้อ Transparent Color ใน Imagelib เป็น $0 เพื่อให้ imagelib รับรู้ว่าพื้นหลังที่เราต้องการให้โปร่งใสนั้นเป็นสีดำ โดยต้องเซตทุกภาพ (คุณอาจเปลี่ยนค่า Transparent Color ให้เป็นแบบอื่น ๆ ได้ตามสีที่ต้องการให้โปร่งใส โดยที่ค่าสีจะเป็นเลขฐาน 16) ดังรูปที่ 3



รูปที่ 3

รูปที่ 4



ลงมือเขียนโปรแกรม

1. เปิดโปรเจ็กต์ใหม่ใน Delphi เปลี่ยนชนิดฟอร์มให้เป็นแบบ TDXForm

2. เพิ่มคอมโปเนนต์ TDXDraw, TDXImmagelist, TDXtimer และ TDXInput ลงในฟอร์มจัดรูปแบบเหมือนรูปที่ 4

3. TDXInput มีคุณสมบัติใน Inspector ที่สำคัญคือ BindinputState และ Enable จะปรากฏอยู่ในหัวข้อ Keyboard, Mouse และ Joystick ถ้าเราต้องการให้ส่วนไหนรับค่าจากผู้ใช้ก็ให้กำหนดคุณสมบัติทั้งสองตัวนี้ให้เป็น True ทั้งหมด ในโปรเจ็กต์ตัวอย่างจะรับค่าจากคีย์บอร์ดและเมาส์ ดังนั้นเราจึงต้องไปกำหนดคุณสมบัติ BindinputState และ Enable ในหัวข้อ Keyboard และ Mouse ให้เป็น True

ตั้งค่าคุณสมบัติ Display ของ TDXDraw ให้เป็น 800x600x16 เพื่อการแสดงผลที่ถูกต้อง

4. ขั้นตอนการทำงานของคอมโปเนนต์ต่าง ๆ ในรูปแบบของการเชื่อมประสาน การวางแผนล่วงหน้าจึงเป็นสิ่งจำเป็น พอจะสรุปขั้นตอนการทำงานของโปรเจ็กต์ประกอบกับโค้ดได้ดังต่อไปนี้

DelphiX กับ Delphi ตอนที่ 3

ลิสติงที่ 1

```
procedure TMainForm.FormCreate(Sender: TObject);
begin
n:=0;
//ตัวแปร n กำหนดค่าไว้สำหรับการวนสุ่มค่าการแสดงรูปภาพของ TDXTimer
ImageList.Items.LoadFromFile(ExtractFilePath(Application.ExeName)+'mos.dxg');
//โหลดแฟ้ม Micro.dxg ขึ้นมาไว้ในหน่วยความจำ
ImageList.Items.MakeColorTable;
DXDraw.ColorTable := ImageList.Items.ColorTable;
DXDraw.DefColorTable := ImageList.Items.ColorTable;
//สร้างตารางสี จากแฟ้มภาพที่โหลดขึ้นมา
xpic:=random(Maxpict);
xpic2:=random(Maxpict);
//สุ่มรูปภาพที่จะนำมาแสดงในครั้งแรก xpic สำหรับยุงตัวที่ 1 xpic2 สำหรับยุงตัวที่ 2
with Sprite do
begin
X := DXDraw.Width div 2-ImageList.Items[xpic].Width div 2;
Y := DXDraw.Height div 2-ImageList.Items[xpic].Height div 2;
end;
//คำนวณหาตำแหน่งที่เหมาะสมในการแสดงรูปภาพของยุงตัวที่ 1
with sprite2 do
begin
X2 := Random(DXDraw.Width-ImageList.Items[xpic2].Width);
Y2 := Random(DXDraw.Height-ImageList.Items[xpic2].Height);
IncX2 := Random(MaxSpeed)+1;
IncY2 := Random(MaxSpeed)+1;
end;
//คำนวณหาตำแหน่งที่จะแสดงภาพยุงตัวที่ 2 โดยการสุ่มตำแหน่ง
end;
สำหรับตัวแปรที่ต้องกำหนดไว้ในช่วงต้นของโปรแกรมมีดังนี้
const
Maxpict = 3;
//นำไปใช้ในการสุ่มรูปภาพ (ซึ่งเตรียมไว้ในแฟ้ม Mos.dxg ทั้งหมด 3 ภาพ)
Maxspeed=10;
//นำไปใช้ในการสุ่มการแสดงตำแหน่งของรูปภาพ ยุงตัวที่ 2

type
TSprite = record
X,x2,Y,y2,IncX,IncY: Longint;
end;
type
TSprite2 = record
x2,y2,IncX2,IncY2: Longint;
end;
//สร้างวัตถุ Tsprite ใช้ดำเนินการกับรูปภาพที่จะแสดง ในที่นี้จะมี 2 Tsprite

public
xpic,xpic2,n:integer;
//xpict,xpic2 เป็นตัวแปรเก็บหมายเลขรูปภาพที่ได้จากการสุ่ม ส่วน n ได้อธิบายไปแล้ว
private
Sprite:TSprite;
Sprite2:TSprite2;
//ตัวแปร Sprite สร้างจากวัตถุ Tsprite ที่เคยสร้างไว้แล้ว ใช้จัดการกับรูปภาพที่จะแสดง
```

ลิสติงที่ 2

```
procedure TMainForm.DXDrawInitialize(Sender: TObject);
begin
DXTimer.Enabled := True;
end;
```

ลิสติงที่ 3

```
procedure TMainForm.DXTimerTimer(Sender: TObject; LagCount: Integer);
begin
if not DXDraw.CanDraw then exit;
DXDraw.Surface.Fill(0);
//ระบายสีใหม่ให้มีพื้นเป็นสีดำ

DXInput.Update;
// ตรวจเช็กการทำงานของ TDXInput ซึ่งรับค่าจากคีย์บอร์ดและเมาส์จากผู้ใช้
begin
with Sprite2 do
begin
Inc(X2, IncX2);
Inc(Y2, IncY2);
if X2<0 then IncX2 := Random(MaxSpeed)+1;
if X2>DXDraw.Surface.Width-ImageList.Items[xpic2].Width then IncX2 := -Random(MaxSpeed)-1;

if Y2<0 then IncY2 := Random(MaxSpeed)+1;
if Y2>DXDraw.Surface.Height-ImageList.Items[xpic2].Height then IncY2 := -Random(MaxSpeed)-1;
//สุ่มตำแหน่งของรูปภาพ ยุงตัวที่ 2 และตรวจเช็กการล้นขอบของการแสดงยุงตัวที่ 2 ทั้งแนวนอนและแนวตั้ง

end;
if n=10 then
begin
xpic:=random(Maxpict);
xpic2:=random(Maxpict);
//สุ่มรูปภาพที่แสดงของยุงตัวที่ 1 และ 2 เมื่อถึงระยะเวลาที่กำหนด (10 Ticks)
n:=0;
end;
n:=n+1;
with Sprite do
begin
//นำค่าจาก TDXInput มาตั้งตำแหน่งแกน X และ Y ของยุงตัวที่ 1 ใหม่
//isLeft, isRight, isUp, isDown คือค่าของคีย์บอร์ดและเมาส์ที่ TDXInput รับเข้ามา

if isLeft in DXInput.States then Dec(X, 4);
if isRight in DXInput.States then Inc(X, 4);
if isUp in DXInput.States then Dec(Y, 4);
if isDown in DXInput.States then Inc(Y, 4);

if X<0 then X := 0;
if X>DXDraw.Surface.Width-ImageList.Items[xpic].Width then X := DXDraw.Surface.Width-ImageList.Items[0].Width;

if Y<0 then Y := 0;
if Y>DXDraw.Surface.Height-ImageList.Items[xpic].Height then Y := DXDraw.Surface.Height-ImageList.Items[0].Height;
//ตรวจเช็กการล้นขอบของการแสดงยุงตัวที่ 1 ทั้งแนวนอนและแนวตั้ง
end;

with Sprite do
ImageList.Items[xpic].Draw(DXDraw.Surface, X, Y, 0);
end;
with Sprite2 do
ImageList.Items[xpic2].Draw(DXDraw.Surface, X2, Y2, 0);
//กำหนดการแสดงรูปภาพในตำแหน่งใหม่ของยุงตัวที่ 1 และ 2
```

→

DelphiX กับ Delphi ตอนที่ 3

```
with DXDraw.Surface.Canvas do
begin
Brush.Style := bsClear;
Font.Color := clWhite;
Font.Size := 12;
Textout(10 , 10, 'โปรดกดแป้นลูกศรเลื่อนขึ้นเลื่อนลง');
Font.Size := 18;
Font.Color := clYellow;
if isLeft in DXInput.States then Textout(10 , 50, 'ซ้าย');
if isRight in DXInput.States then Textout(10 , 50, 'ขวา');
if isUp in DXInput.States then Textout(10 , 50, 'ขึ้น');
if isDown in DXInput.States then Textout(10 , 50, 'ลง');
Release;
end;

//จะแสดงผลแล้วนะ
DXDraw.Flip;
//OK
end;
```

```
procedure TMainForm.DXDrawFinalize(Sender: TObject);
begin
DXTimer.Enabled := False;
end;
```

ลิสติงที่ 4

```
procedure TMainForm.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
//ถ้าผู้ใช้กดแป้น ESC ให้ยุติการทำงาน
if Key=VK_ESCAPE then
Close;
//ถ้าผู้ใช้กดแป้น Home ให้สุ่มรูปภาพอีกครั้ง (เป็นการแทรกแซงการทำงานของ TDXtimer โดยผู้ใช้)
if Key=VK_HOME then
xpic:=random(Maxpict);
//ถ้าผู้ใช้กดแป้น Alt+Shift+Enter ให้เปลี่ยนโหมดการแสดงผลจากเต็มจอ ไปสู่วินโดว์ธรรมดา หรือจากวินโดว์ธรรมดาให้เป็นเต็มจอ
if (ssAlt in Shift) and (Key=VK_RETURN) then
begin
DXDraw.Finalize;
if doFullScreen in DXDraw.Options then
begin
RestoreWindow;
DXDraw.Cursor := crDefault;
BorderStyle := bsSizeable;
DXDraw.Options := DXDraw.Options - [doFullScreen];
end else
begin
StoreWindow;

DXDraw.Cursor := crNone;
BorderStyle := bsNone;
DXDraw.Options := DXDraw.Options + [doFullScreen];
end;
//ถ้าไม่มีอะไรเปลี่ยนแปลง หรือเปลี่ยนเสร็จแล้ว กลับสู่การทำงานของ Event Initialize ของ TDXDraw อีกครั้ง
DXDraw.Initialize;
end;
end;
```

ลิสติงที่ 5

DelphiX กับ Delphi ตอนที่ 3

ลิสติงที่ 6

```
unit Main;

interface

uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ExtCtrls, Menus, DXInput, DXClass, DXDraws;

const
Maxpict = 3;
Maxspeed=10;

type
TSprite = record
X,x2,Y,y2,IncX,IncY: Longint;
end;
type
TSprite2 = record
x2,y2,IncX2,IncY2: Longint;
end;

TMainForm = class(TDXForm)
DXDraw: TDXDraw;
ImageList: TDXImageList;
DXTimer: TDXTimer;
DXInput: TDXInput;
procedure FormKeyDown(Sender: TObject; var Key: Word;Shift: TShiftState);
procedure DXDrawFinalize(Sender: TObject);
procedure DXDrawInitialize(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure DXTimerTimer(Sender: TObject; LagCount: Integer);

public
xpic,xpic2,n:integer;
private
Sprite:TSprite;
Sprite2:TSprite2;
end;

var
MainForm: TMainForm;

implementation

uses MMSystem;

{$R *.DFM}


procedure TMainForm.FormCreate(Sender: TObject);
begin
n:=0;
ImageList.Items.LoadFromFile(ExtractFilePath(Application.ExeName)+'mos.dxg');

ImageList.Items.MakeColorTable;
DXDraw.ColorTable := ImageList.Items.ColorTable;
DXDraw.DefColorTable := ImageList.Items.ColorTable;
xpic:=random(Maxpict);
xpic2:=random(Maxpict);
with Sprite do
begin
X := DXDraw.Width div 2-ImageList.Items[xpic].Width div 2;
Y := DXDraw.Height div 2-ImageList.Items[xpic].Height div 2;
end;
with sprite2 do
begin
X2 := Random(DXDraw.Width-ImageList.Items[xpic2].Width);
Y2 := Random(DXDraw.Height-ImageList.Items[xpic2].Height);
IncX2 := Random(MaxSpeed)+1;
IncY2 := Random(MaxSpeed)+1;
end;
end;

procedure TMainForm.DXDrawInitialize(Sender: TObject);
begin
DXTimer.Enabled := True;


end;

procedure TMainForm.DXDrawFinalize(Sender: TObject);
begin
DXTimer.Enabled := False;

end;

procedure TMainForm.DXTimerTimer(Sender: TObject; LagCount: Integer);
begin
if not DXDraw.CanDraw then exit;

DXDraw.Surface.Fill(0);

DXInput.Update;
begin
with Sprite2 do
begin
Inc(X2, IncX2);
Inc(Y2, IncY2);
if X2<0 then IncX2 := Random(MaxSpeed)+1;
if X2>DXDraw.Surface.Width-ImageList.Items[xpic2]
.Width then IncX2 := -Random(MaxSpeed)-1;

if Y2<0 then IncY2 := Random(MaxSpeed)+1;
if Y2>DXDraw.Surface.Height-ImageList.Items[xpic
2].Height then IncY2 := -Random(MaxSpeed)-1;
end;
if n=10 then
begin
xpic:=random(Maxpict);
xpic2:=random(Maxpict);
n:=0;
end;
n:=n+1;
with Sprite do
begin
if isLeft in DXInput.States then Dec(X, 4);
if isRight in DXInput.States then Inc(X, 4);
if isUp in DXInput.States then Dec(Y, 4);
if isDown in DXInput.States then Inc(Y, 4);

if X<0 then X := 0;
if X>DXDraw.Surface.Width-ImageList.Items[xpic]
.Width then X := DXDraw.Surface.Width-ImageList.Items[0].Width;

if Y<0 then Y := 0;
if Y>DXDraw.Surface.Height-ImageList.Items[xpic].Height then Y :=
DXDraw.Surface.Height-ImageList.Items[0].Height;
end;
with Sprite do
ImageList.Items[xpic].Draw(DXDraw.Surface, X, Y, 0);
end;
with Sprite2 do
ImageList.Items[xpic2].Draw(DXDraw.Surface, X2, Y2, 0);
```

→

```
with DXDraw.Surface.Canvas do
begin
Brush.Style := bsClear;
Font.Color := clWhite;
Font.Size := 12;
Textout(10 , 10, 'โปรดกดแป้นลูกศรเลื่อนขึ้นเลื่อนลง');
Font.Size := 18;
Font.Color := clYellow;
if isLeft in DXInput.States then Textout(10 , 50, 'ซ้าย');
if isRight in DXInput.States then Textout(10 , 50, 'ขวา');
if isUp in DXInput.States then Textout(10 , 50, 'ขึ้น');
if isDown in DXInput.States then Textout(10 , 50, 'ลง');
Release;
end;

DXDraw.Flip;
end;


procedure TMainForm.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if Key=VK_ESCAPE then
Close;
if Key=VK_HOME then
xpic:=random(Maxpict);

if (ssAlt in Shift) and (Key=VK_RETURN) then
begin
DXDraw.Finalize;

if doFullScreen in DXDraw.Options then
begin
RestoreWindow;

DXDraw.Cursor := crDefault;
BorderStyle := bsSizeable;
DXDraw.Options := DXDraw.Options - [doFullScreen];
end else
begin
StoreWindow;

DXDraw.Cursor := crNone;
BorderStyle := bsNone;
DXDraw.Options := DXDraw.Options + [doFullScreen];
end;

    DXDraw.Initialize;
  end;
end;
```

4.1 เมื่อเปิดฟอร์มขึ้นมาครั้งแรก ตัวที่ต้องทำงานเป็นอันดับแรกก็คือ TDXimagelist ซึ่งต้องโหลดแฟ้มรูปภาพและกำหนดตารางสีเพื่อส่งไปที่ TDXDraw พร้อมทั้งการสุ่มหารูปภาพที่ต้องนำมาแสดงในหน้าจอแรกและคำนวณหาตำแหน่งที่เหมาะสมที่จะแสดงรูปภาพ ดังนั้นใน Event Create ของฟอร์มหลัก (TmainForm) จึงต้องมีโค้ดดังลิสติงที่ 1

4.2 เมื่อขั้นตอนที่ 1 ผ่านไป การทำงานก็จะถูกโยนไปที่ Event initialize ของ TDXDraw ซึ่งมีโค้ดดังลิสติงที่ 2 นั่นคือตัว TDXTimer จะถูกสั่งให้เริ่มทำงาน

4.3 โค้ดใน Event Timer ของคอมโปเนนต์ TDXTimer จะถูกสั่งการในลำดับถัดมา (ที่จริง Event Activate จะทำงานก่อน แต่ในที่นี้ไม่ได้สั่งการอะไรใน Event ดังกล่าว) ดังลิสติงที่ 3

4.4 เมื่อเสร็จสิ้นการทำงานก็จะกลับไปสู่ Event Finalize ของคอมโปเนนต์ TDXDraw ดังลิสติงที่ 4

นั่นคือการกำหนดให้การทำงานของ TDXTimer สิ้นสุดลง เพื่อไปตรวจเช็กการกดคีย์บอร์ดใน Event Keydown ของฟอร์มหลัก ดังลิสติงที่ 5

ในที่สุดเมื่อพิมพ์โค้ดจนจบกระบวนการ ซึ่งโค้ดเต็ม ๆ ของโปรเจ็กต์แสดงลิสติงที่ 6 ครับ

ซึ่งเมื่อ Run โปรแกรมก็จะทำงานโดยแสดงภาพจากการสุ่มออกมาโดยจะเปลี่ยนไปเรื่อย ๆ ตามช่วงเวลา ส่วนประกอบของการแสดงจะเพิ่มเติมจากบทความตอนที่แล้ว ครั้งนี้จะเป็นการแสดงภาพสองส่วน คือยุงตัวที่หนึ่งที่จะมีการเคลื่อนไหวเปลี่ยนอริยาบทตามการสุ่มภาพ เคลื่อนที่ตามการรับค่าจากคีย์บอร์ดและเมาส์ ส่วนยุงตัวที่สองจะมีการเคลื่อนไหวเปลี่ยนอริยาบทตามการสุ่มภาพเช่นกัน แต่การเคลื่อนที่จะเป็นไปตามการสุ่มตำแหน่ง ภาพที่แสดงจะดูนุ่มนวลเหมือนกับมีชีวิตจิตใจและเป็นอิสระจากการควบคุมทั้งปวง

สำหรับการเปลี่ยนอิริยาบทอาจดูกระตุกไปบ้าง เนื่องจากใช้รูปภาพเพียง 3 ภาพ ถ้าจะให้ดูต่อเนื่องและเป็นธรรมชาติมากกว่านี้คงต้องเพิ่มรูปแสดงอิริยาบทให้มีความละเอียดมาก (ภาพ) ขึ้น

บทสรุป

สำหรับประโยชน์ของความรู้จากบทความตอนนี้ก็เป็นความรู้พื้นฐานการเขียนเกมที่มีการเคลื่อนไหวในแบบ 2 มิติโดยการสลับรูปภาพทำให้เกิดการหลอกตาผู้ใช้ให้เห็นว่าวัตถุเกิดการเคลื่อนไหว ซึ่งเกมยอดนิยมหลาย ๆ ตัวก็ใช้เทคนิคนี้ในการแสดงรูปภาพ ไม่ว่าจะเป็น Pac-Man, Prince of persia ภาค 1-2, Mario หรือเกมแนววางแผนหลายตัว รูปแบบเกม 2 มิติเชื่อว่ายังจะได้รับความนิยมอีกนาน สำหรับผู้เล่นที่ชอบความง่ายสบายตา ไม่ทำให้เวียนศีรษะหรือตาลายเหมือนกับเกมสามมิติ

สำหรับบทความตอนต่อไปจะนำเสนอการเคลื่อนไหวแบบสองมิติที่เข้มข้นและดูนุ่มนวลกว่าตอนนี้ และจะเพิ่มความยากตามงานที่ทำ ส่วนคุณผู้อ่านที่อยากจะเขียนเกมสามมิติจริง ๆ จัง ๆ ต้องขอให้อดใจรอเพราะผู้เขียนตั้งใจจะนำเสนอในตอนต่อ ๆ ไปครับ